REMARKS

This Amendment and Response is responsive to the Examiner's action mailed July 11, 2002. The Office Action has been carefully considered. Reconsideration of the claims as amended, and in view of the following remarks, is respectfully requested.

A. Summary of Examiner's Action and Applicant's Response

1. Indefinite Rejection

The examiner rejected claim 2 to under 35 USC 112 as being indefinite regarding the term "sub-sampling frames" in relation to the term "sub-sampling pixels" as cited in claim 1. These two terms do not have the same meaning. Applicants respectfully traverse the rejection and requests reconsideration for the reasons discussed below.

2. Prior Art Rejections

Claims 1-20 were rejected as unpatentable under 35 U.S.C 103(a) as being obvious in view of Hoffert (US 5,047,853) in view of Brusewitz (US 6,384,862). Applicants respectfully traverse the rejection and requests reconsideration for the reasons discussed below.

B. Applicants Terminology

1. Sampling and Sub-sampling

Applicants used two separate claim terms "sub-sampling pixels" and "sub-sampling frames". As explained in the specification (including the 09/312,922 application), when video is converted from analog to digital, the infinite analog signal must be "*sampled*" to create a finite number of digital values (page 3, Video Signals). Standard video is normally *sampled* at a frame rate (temporal resolution) of 30 frames per second (images per second), at a width (horizontal resolution) of 640 pixels per line, at a height (vertical resolution) of 480 pixels per line, and a pixel bit depth (color resolution or color depth) of 24 or 32 bits per pixels. The width and height combined together also are typically referred to in the art as dimensions (spatial resolution). When a subset of the original samples is taken, in any of these various resolutions, from the original sample set, the taking of the subset is referred to as "*sub-sampling*". Thus frames, width, height, and pixel bit depth can all be

independently *sub-sampled*. For example, if an original sample were taken at 30 frames per second, a sub-sampling of every third frame would result in a frame sub-sample of 10 frames per second. Further, for example, if an original sample were taken at 640 pixels per line, a sub-sampling of every fourth pixel would result in a 160 pixel width. Further, for example, if an original sample were taken at 480 pixels per column, a sub-sampling of every sixth pixel would result in an 80 pixel height. Further, for example, if an original sample were taken at 32 bits per pixel, a sub-sampling of 6 bits of each of the three color components would result in an 18 bit color depth.

Further, an image may be sub-sampled by selecting a portion of an image, as disclosed in co-pending application 09/433,978, filed 4 November 1999. When a portion of an image is sampled the subset has the same spatial resolution as the original but area of the sub-sample is less than the area of the original sample. For consistency this could be referred to as area sub-sampling.

In the specification of the subject invention, applicants described a sub-sampling step 110 (Fig 1) that comprises one or more of the sub-sampling described above (page 11, regarding Fig 1). In reference to Fig 2A-2H, applicants further described and provide example embodiments of sub-sampling pixel bit depth (page 11-13 in relation to Fig 2A-2H) to obtain an 8 bit pixel 299 value (Fig 2H), from an original sample of either 32 bits (Fig 2A) or 24 bits (Fig 2B-2G). Further, applicants state, "The encode table 300 reduces the 8 bit value 299 to a 5 bit code" (page 14, line 1). Applicants also disclosed other embodiments of the invention that further sub-sample the 8-bit pixel value to a 5 bit pixel value. The specification states that any "source resolution less than or equal to five bits ... could be encoded by shifting bits as explained below." (page 19 line 1-2) and that "Fig 8 illustrates an alternate embodiment of this invention where tables are not used for encoding and decoding. Instead, the high order 5 bits 810-818 of an 8 bit pixel 800 are shifted to the right by 3 bit positions to form a five bit sample 830." (page 19, line 5-6). These are examples of sub-sampling pixels to obtain an 8 bit pixel 299 sub-sample from larger 32 bit pixel value 200 or a 24 bit pixel value 210, and obtaining a 5 bit sample 830, which is clearly an 5 bit *sub*-sample taken from an 8 bit pixel 800.

2. Sub-sampling Frames

In the subject invention, frames are encoded one frame, or image, at a time. Frames may be sub-sampled by skipping frames that are in the original frame set. In claim 2, the phrase "rate of

sub-sampling frames" refers to the number of frames per seconds, or the number of images per second, that are actually processed by the compression method.

3. Sub-sampling Pixels

The phrase "sub-sampling pixels from an image" comprises "selecting pixels values for encoding" (page 11, line 22) and includes "sub-sampling of an 8 bit pixel value" (page 11, last sentence). When pixels values are selected for encoding from a sample frame (or image), width sub-sampling, height sub-sampling, and area sub-sampling each could occur, individually or in combination, resulting in a subset of pixels from the original image. The bit depth of each pixel in the subset of pixels may then be sub-sampled resulting in a subset of bits from each selected pixel.

Discussion of Cited Prior Art, i.e., Hoffert and Brusewitz

1. Overview

The combination of Hoffert and Brusewitz are the basis for all of the prior art rejections.

2. Hoffert is a Block-based, Color Compression Method

Hoffert is, as entitled, a method for compressing and decompressing color video data that used luminance partitioning. Specifically, Hoffert breaks an image into blocks of pixels, such as 4 x 4 pixel blocks. The blocks are then encoded with various formats of varying length. Hoffert uses a two-bit code to identify the type of encoding for each block. One code uses an 8-bit block run length. In all cases, color values are coded with 14 bits of color depth. Each of the elements of Hoffert method is very different than that of applicants' invention.

3. Hoffert is for Color Video and is Not Effective for Grayscale Content

Hoffert is designed for color video. Hoffert's method would not be as effective as applicants' invention for handling video with grayscale content. Hoffert method would be ineffective with only five bits as the number of bits selected from each pixel as recited by applicants' claim 4 or where the codes are five bits in length as recited by applicants' claim 11.

4. Hoffert Uses Blocks of Pixels rather than Pixels

Hoffert's methods require procession blocks of pixels rather than individual pixels. In order the process blocks of pixels a plurality of lines must be available to the compressor and the method must backtrack to access pixels that were skipped when a prior block was being processed. Applicants' method is much simpler and efficient.

Hoffert's block based method results in substantial differences in several elements throughout the method.

Specifically, Hoffert's use of blocks is not the same applicants' use of pixels for coding. Hoffert fails to teach a circuit "for counting repeated instances of *a pixel value* when scanning said plurality of pixels" as recited in applicants' claim 15 but instead counts instances of repeated *blocks*.

Further, Hoffert's coding describes blocks rather than instances of pixel values.

5. Hoffert Fails to Select a Code based on Pixels or By Using an Encode Table

Hoffert's fails to "select a code based on a number bits from each pixel" as recited in applicants' claim 1 and fails teach a "encode table" as recited in applicant's claim 5. Hoffert's two bit code (Hoffert's Fig 2) is based on the type of the block, not based on a number of bits from each pixel. Hoffert does not teach an encode table as taught by applicant's Fig 3A, 3B, 10A or 10B.

6. Hoffert Fails to Teach an Encoding Table or Encryption

As stated above Hoffert fails to teach an encoding table as taught by applicants invention. The addition feature of encryption is also lacking. Without the encryption feature, applicants' encoding tables are arranged such that there is a direct correlation between the intensity level and the position in the table. As taught by applicants' on page 20 in relation to Fig 10A to 10C, "both tables are rearranged such that the direct correlation between the intensity level and the position in the table is broken". Further "the encoded data 140 (Fig 1) becomes a cipher and has a higher level of security." Hoffert's two-bit compression type codes in Fig 2 are not the same as applicants' five-bit codes that are related to the values of the pixels being encoded. Hoffert's codes have arbitrary meaning. Hoffert clearly lacks the essential teaching that the code tables can be used for encryption.

7. Hoffert's Run Length is not the Same as Applicants' Run Length

Hoffert's an 8-bit run length that would not be operative or efficient in applicants' invention with a 7 bit run length. As explained above, the respective run lengths are measuring different things. In fact, in Hoffert's decompression device, a 16 pixel decrementer 108 is taught, but it is always set to the same number 16, which is the fixed size of the 4 x 4 block. Clearly, Hoffert's run length decrementer 107 is taught to have a different function than Hoffert's 16 pixel decrementer 108 (which is a 4 bit counter). Both of these decrementers are substantially different in structure, function, and method than applicants' run length. In this regard Hoffert teaches away from the subject invention.

8. Hoffert is More Complex and Applicants Eliminate Steps/Elements

Hoffert's block-based method requires multiple steps or device elements to group pixels into blocks and then process blocks while encoding, and processing blocks and then breaking blocks into pixels while decoding. Applicants' method eliminates these unnecessary steps or elements and therefore Hoffert to not make Applicants method obvious.

9. Hoffert Fails to Teach a Streaming Buffer

Hoffert fails to teach "streaming" an "encoded data buffer" as cited by applicants' claim. It can be assumed that Hoffert sends his codes un-buffered. In fact, Fig 15 indicates that the "encoded data" is feed directly to the MUX 149 without buffering. In this regard, Hoffert teaches away from the subject invention.

10. Hoffert Fails to Teach Sub-sampling Pixels from an Image

As stated by the office action, "Hoffert '853 fails to explicitly teach sub-sampling pixels from an image".

11. Hoffert Teaches Away from the Subject Invention

As mention above. Hoffert teaches away from the subject invention in a number of substantial elements. Specifically, Hoffert teaches:

a. A method for compressing color, rather than grayscale content.

b. Block-based, rather than pixel based processing

c. Block-based run length, rather than pixel based run length.

d. Un-buffered passing of encoded data, rather than buffered streaming of encoded data.

12. Reliance on Brusewitz

In this office action, Brusewitz is relied on to teach (in combination with Hoffert):

a) "sub-sampling image" (sub-sampler 20 [sic])

b) buffers 22 and 30

c) storage 34

Brusewitz subsampling step, flow regulating buffers, and image storage are not the same or equivalent to those of applicants' pixel bit depth sub-sampling, encoded data buffer, and storage medium, as will be explained below.

13. Brusewitz's Imaging System

Brusewitz is an imaging system with an encoder in a sending device and decoder in a receiving device. Brusewitz increases viewer control by including a backchannel whereby a viewer may adjust spatial and temporal resolution of the image being sent. (Brusewitz, Abstract)

14. Brusewitz Fails to Clearly Teach Pixel Bit Depth Sub-sampling

Brusewitz teaches a subsampler 18, which "determines pixel values representing the captured video image at a particular spatial resolution, i.e., pixels per line and lines per image, and temporal resolution, i.e., images per second." Brusewitz however fails to teach pixel bit depth sub-sampling that is a substantial element of applicants' invention.

15. Brusewitz Fails to Clearly Teach Applicants' Buffers

Brusewitz teaches use of a buffer in the sending device 12 "the primary purpose of buffer 22 is to regulate the flow of data from the encoder 20 and forward that data at a fixed rate across a transmission channel 26 to a receiver device 28, particularly, to another buffer 30 therein, which like buffer 22 acts as a reservoir storing the data and regulating its use." Applicants' buffer is used to hold variable length encoded data that represents a single image. In one embodiment of applicant's invention, the buffer is sent as soon as it is complete without regard to regulating flow "at a fixed rate across a transmission channel." Brusewitz buffers are different than applicants' buffer in structure, function, and purpose. Thus Brusewitz teaches away from applicants' buffers.

Further, applicants' "encoded data buffer" contains data encoded with the method of applicants' invention, because Brusewitz did not anticipate the method of applicants' invention, Brusewitz buffers cannot have contained applicants' codes. Instead, Brusewitz taught compression methods including MPEG, H.261, H.262, and H.263 that have their own codes. In this way Brusewitz teaches away from the subject invention.

However, applicants anticipate that applicants' "encoded data buffers" could also be passed through a communications channel comprising Brusewitz's buffers for regulating data flow and use without affecting the scope of applicants' invention.

16. Brusewitz Teaches Away from Applicants' Storage

Brusewitz teaches use of "an image storage device 34, the contents of which may be continuously displayed on a video display 36, the circuitry of which is understood in the art." One skilled in the art would understand this image storage to be a video RAM that is continuously accessed by a display device to display a single image (or a single video frame). In contrast

applicants' storage medium is used to hold an encoded video signal that comprises a plurality of encoded data buffers. Brusewitz image storage is different than applicants' storage medium in content, structure, function, and purpose. Thus Brusewitz teaches away from applicants' storage medium.

Further, applicants' storage medium contains encoded data. Brusewitz image storage clearly contains the output of the decoder that is the reconstituted image, rather than the encoded data as required by applicants' claims 6 and 7. Brusewitz image lacks the essential elements of applicants' storage medium.

17. Brusewitz Away from Applicants' Lossless Compression

Brusewitz teaches compression methods including MPEG, H.261, H.262, and H.263. One skilled in the art would have understood these methods to block-based, lossy compression methods that are substantially different than the applicants' methods. These methods also comprise a quantization parameter that is a measure of distortion allowed by the lossy method.

A major objective of applicants' invention is to provide the lossy distortion of these other methods. In this regard, Brusewitz teaches away from applicants' invention and applicants' invention teaches away from this aspect of Brusewitz.

18. Brusewitz Teaches away from the Subject Invention

Brusewitz teaches away from the subject invention. Specifically, Brusewitz teaches away from applicants' encoded data buffer and storage medium. Brusewitz teaches toward compression methods that utilize block-based, lossy methods and away from applicants' pixel based, lossless methods. Brusewitz's method of changing quantization of the various compression methods including MPEG and other DCT-based compression method teaches away from applicants' invention.

19. Hoffert and Brusewitz Combined Do Not Teach the Invention

Hoffert does not teach the elements of applicants' invention but instead teaches away from them. Brusewitz does not teach the elements of applicants' invention but instead teaches away from them. If Hoffert's elements were combined with Brusewitz's element applicants' invention would not have been taught.

20. Hoffert and Brusewitz Do Not Contain Any Justification To Support Their Combination, Much Less in the Manner Proposed

With regard to the proposed combination of Hoffert and Brusewitz, it is well known that in order for any prior art references themselves to be validly combined for use in a prior art 103 rejection, the *references themselves* (or some other prior art) must suggest they be combined (In re Sernaker, 217 U.S.P.Q. 1, 5 (C.A.F.C. 1983)). "There must be some reason for the combination other than hindsight gleaned from the invention itself... Something in the prior art must suggest the desirability and thus the obviousness of making the combination." (Uniroyal, Inc. v. Rudkin-Wiley Corp., 5 U.S.P.Q. 2d 1434 (C.A.F.C. 1988).

There are no teachings in either reference to combine. Hoffert did not suggest that it would be desirable to provide Brusewitz's remote interface and back channel for controlling compression parameters. Hoffert did not suggest the use of Brusewitz's temporal and spatial subsampling, flow regulating buffers, or image storage. Brusewitz was filed long after Hoffert was published and failed to teach the use of Hoffert's methods. Brusewitz does not teach that it would be desirable to use any of the details of Hoffert's compression method.

21. Hoffert and Brusewitz Are Individually Complete

Each reference is complete and functional in itself, so there would be no reason to use parts from or add or substitute parts of any reference.

22. Hoffert and Brusewitz Take Different Approaches and Teach Away for Each Other

The references take mutually exclusive paths and reach different solutions to a similar problem. Since they teach away from each other, it would not be logical to combine them.

23. Hoffert and Brusewitz Are From Different Fields

Both references are from different technical fields as evidenced by different US classes and fields of search. It would not have been obvious to one skilled in either of these two fields to combine the two to form applicants' invention.

24. Combining Hoffert and Brusewitz is Improper

Applicants therefore submit that combining Hoffert and Brusewitz is not legally justified and is therefore improper. Thus applicants submit that the rejection on these references is also improper and should be withdrawn.

25. Summary

In summary, Hoffert does not teach all of the elements and limitations of claims 1-20, and thus, does not make obvious claims 1-20 of the present invention. Brusewitz by itself does not

teach all the elements and limitations of claims 1-20 does make obvious claim 11 of the present invention. It is improper to combine the teachings of Hoffert and Brusewitz to regarding claims 1-20. Applicants' invention produces new and unexpected results.

C. Applicant's Claims Are Patentable Over the Cited References for Obviousness

Claims 1, 15

Applicants respectfully disagree with the office action's conclusion that Hoffert teaches "selecting a code based on a number of bits from each pixel selected from pixels (i.e. fig 2)." As stated above Hoffert Fig. 2 shows two-bit codes based on analysis of the contents of 4 x 4 blocks of pixels. Hoffert lacks the essential function of selecting a code based on a number of bits from each pixel. Hoffert teaches instead a block-based method of coding. In the preferred embodiment, applicants' code is a five-bit code as shown in Fig 5C, 6, and 8.

Applicants respectfully disagree with the office action's conclusion that Hoffert teaches "run-length encoding repeated instances" of said code. As stated above, Hoffert's run lengths are not the same as applicants' run lengths. Hoffert's instances are repeated blocks not repeated pixels. Hoffert's codes are not the same or equivalent to applicants codes.

Applicants respectfully disagree with the office action's conclusion that Hoffert teaches "repeating steps [(b) and (c)] until each pixel is encoded in an encoded data buffer". As stated above, Hoffert does not teach an encoded data buffer. Brusewitz does not teach applicants' encoded data buffer even if it did it would improper to combine Hoffert with Brusewitz in the manner required by this claim element. Further, Hoffert repeats steps on a block basis and requires additional step elements that are eliminated by applicants' invention. And further, as explained above Hoffert teaches away from buffering encoded data.

Applicants respectfully disagree with the office action's conclusion that "streaming buffer is an inherent feature necessitated by the digital video processing." As stated above, buffering is not required as data can be processed without a buffer as is taught by Hoffert in relation to Fig 15. Because Hoffert teaches away from using a buffer it would improper to combine Hoffert with Brusewitz in the manner required by this claim element.

Applicants respectfully disagree with the office action's conclusion that "Brusewitz…teaches sub-sampling image." As stated above, Brusewitz lacks the essential

function of sub-sampling pixel bit depth (e.g. selecting 8 bit from a 32 bit pixel) as taught by the subject invention.

Applicants respectfully disagree with the office action's conclusion that "it would have been obvious to one having ordinary skill in the art at the time of the invention to modify the system of Hoffert '853 as taught by Brusewitz '862 for customizing the images to the viewers specifications". For the reasons stated above, Hoffert and Brusewitz should not be combined.

For all these reasons, applicants submit that claims 1 and 15 as written are patentable over the references.

Claims 2, 3, 9, and 10

Claims 2 and 3 add additional limitations on claim 1. Applicants submit that claim 2 and 3 should now be allowable because they are dependent on claim 1 which should now be allowable for the reasons cited above.

Claims 9 and 10 add additional limitations on claim 8. Applicants submit that claim 9 and 10 should now be allowable because they are dependent on claim 8 which should now be allowable for the reasons cited above, and below in relation to claim 8.

For all these reasons, applicants submit that claims 2, 3, 9 and 10 as written are patentable over the references.

Claims 4, 5, 11, 12, and 20

Claims 4 and 5 add additional limitations on claim 1. Applicants submit that claim 4 and 5 should now be allowable because they are dependent on claim 1 which should now be allowable for the reasons cited above.

Claims 11 and 12 add additional limitations on claim 8. Applicants submit that claim 11 and 12 should now be allowable because they are dependent on claim 8 which should now be allowable for the reasons cited above, and below in relation to claim 8.

Claim 20 adds an additional limitation on claim 19. Applicants submit that claim 20 should now be allowable because it is dependent on claim 19 which should now be allowable for the reasons cited above, and below in relation to claim 19.

Further, applicants respectfully disagree with the office action's conclusion that "Hoffert ...teaches that the number of bits is five (i.e. fig 1 of Hoffert '853)." Fig 1 of Hoffert does not five bits but instead shows 14 bits per pixel. As taught by Hoffert the pixel value includes 5 bits of red, five bits of green, and five bits of blue for a total of 14 bits. Fig 1 also clearly shows a two-bit type of encoding in a 16-bit field. Sixteen minus 2 is 14. Hoffert states, "The 14 bits following indicates the color" (Hoffert, Column 4 Lines 24-25). Other references in Hoffert are consistent with this 16-bit code with 14 bits of pixel value (see Fig 2, Fig 9's 16 pixel accumulate 57, Fig 10's 14 bit wide bitstream 100). The office action clearly misunderstands the reference. Hoffert fails to teach a five-bit code.

Further, there is no teaching in the references "whereby said image is an enhanced representation of the original image" as required by the limitation on claim 12. The essential element of image enhancement as taught by applicants' invention is missing from the references.

Further regarding, claims 4 and 11, Hoffert fails to teach selection of the five bits by extracting the five most significant bits of each pixel, which is a necessary element of these claims.

Further regarding claim 5, Hoffert fails to teach the necessary element that the "code is obtained from an encode table".

Further regarding claims 12 and 20, Hoffert fails to teach the necessary elements that the "pixel values are obtained from a decode table" or "a decode table lookup", respectively.

For all these reasons, applicants submit that claims 4, 5, 11, 12, and 20 as written are patentable over the references.

Claims 6 and 7

Claims 6 and 7 as amended add additional limitations on claim 1. Applicants submit that claim 6 and 7 should now be allowable because they are dependent on claim 1 which should now be allowable for the reasons cited above.

Further, applicants respectfully disagree with the office action's conclusion that the combination teaches a series of buffer and storage as claimed by applicants. As explained above, neither applicants' series of encoded data buffers nor storage medium are not taught by either

Hoffert or Brusewitz, either individually or in combination. It would also be improper to combine the references.

For all these reasons, applicants submit that claims 6 and 7 as amended are patentable over the references.

Claim 8

Claim 8 is an independent claim and each of its elements must be taught by one or more prior art reference. For all the reasons stated above, applicants disagree that either reference or the combination of reference teach elements and limitations of claim 8 as specified by applicants.

Further, applicants submit that the Fig 15 mux 149 does not performs the function of "combining" as stated by the office action, but the MUX functions to select from either the original data or the encoded data based on whether or not the Hoffert methods determines that duochrominance-isoluminance problems are present (Hoffert C13 L37 through C14 L56). Applicants do not believe this MUX reference teaches anything about the subject invention.

For all these reasons, applicants submit that claim 8, and its dependent claims, as written are patentable over the references.

Claims 13, 14, and 16

Claim 13 adds an additional limitation on claim 5. Applicants submit that claim 5 should now be allowable because it is dependent on claim 5 which should now be allowable for the reasons cited above.

Further, the limitation in claim 13 that "the lines of the encode table are randomly order forming an encryption table so that the direct correlation between the original values and their respective codes are encrypted" is totally lacking in the references, as explained above.

Claim 14 adds an additional limitation on claim 12. Applicants submit that claim 14 should now be allowable because it is dependent on claim 12 which should now be allowable for the reasons cited above.

Further, the limitation in claim 14 that "the lines of the decode table are order in sequence matching said encryption table so that the correct final pixel values are displayed" is totally lacking in the references, as explained above.

Claim 16 adds an additional limitation on claim 15. Applicants submit that claim 16 should now be allowable because it is dependent on claim 15 which should now be allowable for the reasons cited above.

Further, the limitation in claim 16 that "encoding circuit performs a table lookup to translate said pixel values into encrypted enhancement codes" is totally lacking in the references, as explained above. The essential features of "a lookup table", encryption, and "enhancement" are not taught by either of the references.

For all these reasons, applicants submit that claims 13, 14, and 16 as written are patentable over the references.

Claims 17 and 18

Claims 17 and 18 adds additional limitations on claim 15. Applicants submit that claims 17 and 18 should now be allowable because it is dependent on claim 15 which should now be allowable for the reasons cited above.

Applicants submit that claims 17 and 18 as written are patentable over the references.

Claims 19

The office action only reference the rejection of claim 19 based on the grounds for rejecting claims 8 and 15. Applicants submit that claim 19 should now be allowable for the same reasons that claims 8 and 15 should now be allowable for the reasons cited above.

Applicants submit that claim 19 as written is patentable over the references.

D. Amended Claims

Applicants submit amended claim 6. Claim 6 is amended to be written in dependent form:

a. to avoid the potential argument that an "encoded video signal" is not an article of manufacture; and

b. to provide a antecedent for "said encoded data buffers".

Note that these changes were not made to overcome prior art references. Prior art obviousness rejections of the elements of claim 6 were transversed above, without amending any element.

Applicants submit claim 6 that should be allowable as amended, for the reasons stated above.

E. New Claims

Applicants submit claims 21 through 24 that should be allowable as written. Note that claims 21 through 24 are the same as claims 28 through 31 that were submitted on 17 May 1999 with application 09/312,922 (with the correction of an obvious error in claim 28(c) now claim 21(c)).

In claim 21(c) the phrase "incrementing a repeat counter if the current line number *does not match* the previous line number" is corrected to read "incrementing a repeat counter if the current line number *matches* the previous line number". This correction is supported by the original disclosure in application 09/312,922 as follows:

a. In Fig 4A, wherein if the answer to 408 "Is Line Number same as Previous Line Num?" is "yes" flow continues to step 410 "Increment Repeat Counter".
b. On page 17 line 14-19, "At the step 408, it is determined whether the previous line number for the pixel data is the same as the previous line number... If the line number is the same as the previous line number, the repeat counter value is incremented by one, at step 410."

The addition of these claims is not being made to overcome any prior art rejections, but is being made in response to an office action dated 12/17/2002 in the co-pending '922 application (signed by the same supervisor as this office action, Chris Kelly), indicating that the compression method is a distinct invention from its combination with the video transmissions system, and requiring a restriction.

Please note that these claims were examined in conjunction with an International application PCT/US99/10894 filed on 17 May 1999, and published as W0 99/59472 on 25 November 1999. The International Preliminary Examination Report, dated 17 August 2000, indicated that claims 28-31 (now 21 through 24 herein) meet the PCT criteria from novelty and inventive step.

Claims 21 through 24 have been added. The features of these claims are novel because, as stated various times above, they are not seen, as specifically claimed in the present invention, in the prior art. Specifically, the present invention embodies systems and methods that allow for fast, real time, and high quality compression of video images. The features of these claims are not obvious and produce superior and or unexpected results from what is found in the prior art. Specifically, by improving the effective compression without the introduction of distortion by conventional compression methods, an number of benefits result, including but not limited to being able to receive an image from a video stream over a connection medium in real-time that could not be received without substantially more distortion, space, bandwidth, or cost.

F. Other Amendments to the Specification

In addition to a number of self-explanatory clerical errors that are corrected, the specification is amended to better integrate the specification and terminology used in the application 09/312,922. The two specifications were written by two different authors who described distinct embodiments of the subject invention with two distinct sets of terminology. Now that claims using the second set of terms are being transferred to this application, a table providing a correlation between the two sets of terminology has been added to aid examiner and the public.

Note that this correlation of terms does not necessarily equivalence. For example, the term "5 bit code" of this applicant is not necessarily equivalent to the term "line number" of the '922 application. The "5 bit code" has a size limitation on the code but does not need to be listed on a particular line in table; for example, it could be a "five bit sample 830". The "line number" is not necessarily limited in size but would be a four bit number if the there were 16 lines or a six bit number if there were 64 lines. Thus the term correlation is provided as a high level aid but the terms themselves should be understood based on the respective original specification and their respective embodiments and figures.

Also note that the same term used in this specification in the field of "compression and decompression of video images" is likely to have different meaning in the '922 specification which was written in the field of "video communications systems" and "medical devices". For example in the field of compression the term "image" generally refers to a still image or a single

frame of a video, but in the medical device field the term "image" often refers to the collection of all the frames in a video. Thus the terms themselves should be understood based on the respective original specification. The legal presumption that terms in related applications by the same inventors have the same meaning does not apply in this case.

Finally, the abstract is amended to include the following sentence:

"High levels of effective compression also reduce the storage space requirement for recorded video."

Support for this is found in the original specification in the Summary of the Invention:

"In accordance with the present invention a method of *compression of a video stream* comprises steps of sub-sampling a video frame, determining a code for each pixel, run-length encoding the codes whereby the method can be executed in real time and the compressed representation of codes *saves substantial space on a storage medium* and require substantially less time and bandwidth to be transported over a communications link." (Emphasis added).

RECONSIDERATION REQUESTED

Attached hereto is a marked-up version of the changes made to the specification by the current amendment. The attached page is captioned **"Version with Markings to Show Changes Made"**. Also, a clean copy of the claims with the new claims is attached.

The undersigned respectfully submits that, in view of the foregoing amendments and remarks, the rejections of the claims raised in the Office Action dated July 11, 2002 have been fully addressed and overcome, and the present application is believed to be in condition for allowance. It is respectfully requested that this application be reconsidered, that these claims be allowed, and a Notice of Allowance is respectfully requested. If it is believed that a telephone conversation would expedite the prosecution of the present application, or clarify matters with regard to its allowance, the Examiner is invited to call the undersigned at 408-739-9517.

Respectfully submitted,

Kendyl A. Roman
Phone: 408-739-9517

Sunnyvale, California

Date: January 13, 2003

Application 09/470,566

Version with Markings to Show Changes Made

Patent Application of
Kendyl A. Román
Cyrus J. Hoomani
and
Richard S. Neale
for

TITLE: GENERAL PURPOSE COMPRESSION FOR VIDEO IMAGES (RHN)

CROSS-REFERENCE TO RELATED APPLICATIONS

This application claims priority under 35 U.S.C. § 119(e) of the co-pending U.S. provisional application Serial Number 60/113,276 filed on 1998 December 23, and entitled "METHOD OF IMAGE ENHANCEMENT, COMPRESSION, AND ENCODING of GRAYSCALE IMAGES (ECHOCODEC)." The provisional application Serial Number 60/113,276 filed on 1998 December 23 and entitled "METHOD OF IMAGE ENHANCEMENT, COMPRESSION, AND ENCODING of GRAYSCALE IMAGES (ECHOCODEC)" is also hereby incorporated by reference.

Our co-pending U.S. patent application Serial Number 90/312,922 filed on 1999 May 17, and entitled "SYSTEM FOR TRANSMITTING VIDEO IMAGES OVER A COMPUTER NETWORK TO A REMOTE RECEIVER" ~~claims portions~~describes an embodiment of the invention of this application in combination with ~~claims from~~ our co-pending U.S. provisional application Serial Number 60/085,818 filed on 1998 May 18, and entitled "APPARATUS FOR TRANSMITTING LIVE VIDEO IMAGES OVER A COMPUTER NETWORK TO MULTIPLE REMOTE RECEIVERS." U.S. patent application Serial Number 90/312,922 filed on 1999 May 17, and entitled "SYSTEM FOR TRANSMITTING VIDEO IMAGES OVER A COMPUTER NETWORK TO A REMOTE RECEIVER" is also hereby incorporated by reference.

BACKGROUND – FIELD OF THE INVENTION

This invention relates to data compression, specifically to the compression and decompression of video images.

BACKGROUND–DESCRIPTION OF PRIOR ART

In the last few years, there have been tremendous advances in the speed of computer processors and in the availability of bandwidth of worldwide computer networks such as the Internet. These advances have led to a point where businesses and households now commonly have both the computing power and network connectivity necessary to have point-to-point digital communications of audio, rich graphical images, and video. However the transmission of video signals with the full resolution and quality of television is still out of reach. In order to achieve an acceptable level of video quality, the video signal must be compressed significantly without losing either spatial or temporal quality.

A number of different approaches have been taken but each has resulted in less than acceptable results. These approaches and their disadvantages are disclosed by Mark Nelson in a book entitled The Data Compression Book, Second Edition, published by M&T Book in 1996. Mark Morrision also discusses the state of the art in a book entitled The Magic of Image Processing ~~The Magic of Image Processing~~, published by Sams Publishing in 1993.

Video Signals

Standard video signals are analog in nature. In the United States, television signals contain 525 scan lines of which 480 lines are visible on most televisions. The video signal ~~represent~~ represents a continuous stream of still images, also known a frames, that are fully scanned, transmitted and displayed at a rate of 30 frames per second. This frame rate is considered full motion. A television screen has a 4:3 aspect ratio.

When an analog video signal is digitized, each of the 480 lines are sampled 640 times and each sample is represented by a number. Each sample point is called a picture element, or pixel. A two dimensional array is created that is 640 pixels wide and 480 pixels high. This 640 x 480 pixel array is a still graphical image that is considered to be full frame. The human eye can perceive 16.7 thousand colors. A pixel value comprised of 24 bits can represent each perceivable color. A graphical image made up of 24-bit pixels is considered to be full color. A single second-long full frame, full color video requires over 220 millions bits of data.

The transmission of 640 x 480 pixels x 24 bits per pixel times 30 frames requires the transmission of 221,184,000 millions bits per second. A T1 Internet connection can transfer up to 1.54 millions bits per second. A high speed (56Kb) modem can transfer data at a maximum rate of 56 thousand bits per second. The transfer of full motion, full frame, full color digital video over a T1 Internet connection, or 56Kb modem, will require an effective data compression of over 144:1, or 3949:1, respectively.

Basic Run-length Encoding

An early technique for data compression is run-length encoding where a repeated series of items are replaced with one sample item and a count for the number of times the sample repeats. Prior art shows run-length encoding of both individual bits and bytes. These simple approaches by themselves have failed to achieve the necessary compression ratios.

Variable Length Encoding

In the late 1940s, Claude Shannon at Bell Labs and R.M. Fano at MIT pioneered the field of data compression. Their work resulted in a technique of using variable length codes where codes with low probabilities have more bits, and codes with higher probabilities have fewer bits. This approach requires multiple passes through the data to determine code probability and then to encode the data. This approach also has failed to achieve the necessary compression ratios.

D. A. Huffman disclosed a more efficient approach of variable length encoding known as Huffman coding in a paper entitled "A Method for Construction of Minimum Redundancy Codes," published in 1952. This approach also has failed to achieve the necessary compression ratios.

Arithmetic, Finite Context, and Adaptive Coding

In the 1980s, arithmetic, finite coding, and adaptive coding have provided a slight improvement over the earlier methods. These approaches require extensive computer processing and have failed to achieve the necessary compression ratios.

Dictionary-Based Compression

Dictionary-based compression uses a completely different method to compress data. Variable length strings of symbols are encoded as single tokens. The tokens form an index to a dictionary. In 1977, Abraham Lempel and Jacob Ziv published a paper entitled, "A Universal Algorithm for Sequential Data Compression" in IEEE Transactions on Information Theory, which disclosed a compression technique commonly known as LZ77. The same authors published a 1978 sequel entitled, "Compression of Individual Sequences via Variable-Rate Coding," which disclosed a compression technique commonly known as LZ78 (see U.S. Patent 4,464,650). Terry Welch published an article entitled, "A Technique for High-Performance Data Compression," in the June 1984 issue of IEEE Computer, which disclosed an algorithm commonly known as LZW, which is the basis for the GIF algorithm (see US Patents 4,558,302, 4,814,746, and 4,876,541). In 1989, Stack Electronics implemented a LZ77 based method called QIC-122 (see U.S. Patent 5,532,694, U.S. Patent 5,506,580, and U.S. Patent 5,463,390). The output of a QIC-122 encoder consists of a stream of data, which, in turn consists of tokens and symbols freely intermixed. Each token or symbol is prefixed by a single bit flag that indicates whether the following is a dictionary reference or a ~~plan~~ plain symbol. The definitions for these two sequences are:

(a) plaintext: <1><eight-bit-symbol>

(b) dictionary reference: <0><window-offset><phrase-length>

Windows offsets are encoded as seven bits or eleven bits. These lossless (method where no data is lost) compression methods can achieve up to 10:1 compression ratios on graphic images typical of a video image. While these dictionary-based algorithms are popular, these approaches require extensive computer processing and have failed to achieve the necessary compression ratios.

JPEG and MPEG

Graphical images have an advantage over conventional computer data files: they can be slightly modified during the compression/decompression cycle without affecting the perceived quality on the part of the viewer. By allowing some loss of data, compression ratios of 25:1 have been achieved without major degradation of the perceived image. The Joint Photographic Experts Group (JPEG) has developed a standard for graphical image compression. The JPEG lossy (method where some data is lost) compression algorithm first divides the color image into three color planes and divides each plane into 8 by 8 blocks, and then the algorithm operates in three successive stages:

(a) A mathematical transformation known as Discrete Cosine Transform (DCT) takes a set of points from the spatial domain and transforms them into an identical representation in the frequency domain.

(b) A lossy quantization is performed using a quantization matrix to reduce the precision of the coefficients.

(c) The zero values are encoded in a zig-zag sequence (see Nelson, pp. 341-342).

JPEG can be scaled to perform higher compression ratio by allowing more loss in the quantization stage of the compression. However this loss results in certain blocks of the image being compressed such that areas of the image have a blocky appearance and the edges of the 8 by 8 blocks become apparent because they no longer match the colors of their adjacent blocks. Another disadvantage of JPEG is smearing. The true edges in an image get blurred due to the lossy compression method.

The Moving Pictures Expert Group (MPEG) uses a combination of JPEG based techniques combined with forward and reverse temporal differencing. MPEG compares adjacent frames and for those blocks that are identical to those in a previous or subsequent frame and only a description of the previous or subsequent identical block is encoded. MPEG suffers from the same blocking and smearing problems as JPEG.

These approaches require extensive computer processing and have failed to achieve the necessary compression ratios without unacceptable loss of image quality and artificially induced distortion.

QuickTime: CinePak, Sorensen, H.263

Apple Computer, Inc. released a component architecture for digital video compression and decompression, ~~name~~ named QuickTime. Any number of methods can be encoded into a QuickTime compressor/decompressor (codec). Some popular codec are CinePak, Sorensen, and H.263. CinePak and Sorensen both require extensive computer processing to prepare a digital video sequence for playback in real time; neither can be used for live compression. H.263 compresses in real time but does so by sacrificing image quality resulting in severe blocking and smearing.

Fractal and Wavelet Compression

Extremely high compression ratios are achievable with fractal and wavelet compression algorithms. These approaches require extensive computer processing and generally cannot be completed in real time.

SUMMARY OF THE INVENTION

In accordance with the present invention a method of compression of a video stream comprises steps of sub-sampling a video frame, determining a code for each pixel, run-length encoding the codes whereby the method can be executed in real time and the compressed

representation of codes saves substantial space on a storage medium and require substantially less time and bandwidth to be transported over a communications link. The present invention includes a corresponding method for decompressing the encoded data.

Objects and Advantages

Accordingly, beside the objects and advantages of the method described in our patent above, some additional objects and advantages of the present invention are:

(a) to provide a method of compressing and decompressing video signals so that the video information can be transported across a digital communications channel in real time.

(b) to provide a method of compressing and decompressing video signals such that compression can be accomplished with software on commercially available computers without the need for additional hardware for either compression or decompression.

(c) to provide a high quality video image without the blocking and smearing defects associated with prior art lossy methods.

(d) to provide a high quality video image that suitable for use in medical applications.

(e) to provide some level of encryption so that images are not directly viewable from the data as contained in the transmission.

(f) to provide a method of compression of video signals such that the compressed representation of the video signals is substantially reduced in size for storage on a storage medium.

(g) to enhance electronically generated images by filtering highs and lows.

DRAWING FIGURES

In the drawings, closely related figures have the same number but different alphabetic suffixes.

Fig 1 shows the high level steps of compression and decompression of an image.

Fig 2A to 2H show alternatives for selecting a pixel value for encoding.

Fig 3A shows the encode table of the preferred embodiment.

Fig 3B shows a chart of values corresponding to the sample encode table.

Fig 4A shows the flowchart for the preferred embodiment of the compression method.

Fig 4B shows an image and a corresponding stream of pixels.

Fig 5A to 5C shows the formats for the run-length encoding.

Fig 6 shows a series of codes and the resulting encoded stream.

Fig 7 shows a sample decode table.

Fig 8 shows an alternate method of selecting a five bit code.

Fig 9 shows the flow chart for the preferred embodiment of the decompression method.

Figs 10A to 10C show an encryption key, an encryption table and a decryption table.

Reference Numerals in Drawings

100	compression steps	110	sub-sampling step
120	code lookup step	130	run-length encoding step
140	encoded data	150	decompression steps
160	run-length expansion step	170	value lookup step
180	image reconstitution step	200	32 bit pixel value
202	blue channel	204	green channel
206	red channel	208	alpha channel
210	24 bit pixel value	212	blue component
214	green component	216	red component
220	RGB averaging diagram	222	blue value
224	green value	226	red value
228	averaged value	230	blue selection diagram
232	blue instance	234	green instance
236	red instance	240	selected blue value
250	green selection diagram	260	selected green value
270	red selection diagram	280	selected red value
290	grayscale pixel	292	grayscale blue
294	grayscale green	296	grayscale red
298	selected grayscale value	299	8 bit pixel value
300	encode table		
310	codes	320	line comments
330	minimum values	340	maximum values
360	stepped values	350	chart of values
370	line numbers	400	encode flowchart
402	encode entry	403	encode initialization step
404	get pixel step	405	get value step
406	lookup encoded value step	408	compare previous
410	increment counter step	412	check count overflow

414	new code step	416	check end of data
418	set done	420	counter overflow step
422	check done	428	encode exit
430	image	440	image width
450	image height	460	pixel stream
500	code byte	510	flag bit
520	repeat code	530	count
550	data code	565	data bit 6
570	data bit 5	575	data bit 4
580	data bit 3	585	data bit 2
590	data bit 1	595	data bit 0
610	decimal values	620	first value
622	second value	624	third value
626	fourth value	628	fifth value
630	sixth value	632	seventh value
640	binary code	650	first byte
651	repeat count	652	second byte
653	first code	654	third byte
655	second code	656	fourth byte
657	third code	700	decode table
710	alpha values	720	red values
730	green values	740	blue values
800	8 bit pixel	810	pixel bit 7
812	pixel bit 6	814	pixel bit 5
816	pixel bit 4	818	pixel bit 3
820	pixel bit 2	822	pixel bit 1
824	pixel bit 0	830	5 bit sample
832	sample bit 4	834	sample bit 3
836	sample bit 2	838	sample bit 1
840	sample bit 0	850	ignored bits
860	decompressed pixel	880	3 low order bits
900	decode entry	901	decode initialize step
902	get code step	906	determine type

908	decode lookup step	909	check zero count
910	place pixel step	912	assign counter step
914	reset counter step	916	check length
918	decode exit	1000	encryption key
1010	encryption table	1020	decryption table

Terminology Correlation

Different terminology was used in the specification of application 09/312,922. The two specifications were written by two different authors who described distinct embodiments of the subject invention with two distinct sets of terminology. The following table provides a partial correlation between the two sets of terminology. Note that this correlation of terms does not necessarily mean the terms are equivalent. This high level correlation is provided to aid in understanding similarities and differences between the two specifications. Also note that the same term used in this specification in the field of "compression and decompression of video images" is likely to have different meaning in the 09/312,922 specification which was written in the field of "video communications systems" and "medical devices". For example in the field of compression the term "image" generally refers to a still image or a single frame of a video, but in the medical device field the term "image" often refers to the collection of all the frames in a video. Thus the terms themselves should be understood based on the respective original specification. The legal presumption that terms in related applications by the same inventors have the same meaning does not apply in this case.

Terms Used In This Application		Terms Used in Application 09/312,922
140	encoded data	stream of compressed data
212	blue component	blue scale value
214	green component	green scale value
216	red component	red scale value
300	encode table, encodeTable	encodePallet
	decodeTable	decodePallet
310	codes	compression lookup table 310
	code	line number
320	line comments	documentation 320
350	chart of values	lookup table 350
370	line numbers	line number (column)

430	image	video image 400
460	pixel stream	stream of pixel data 405
500	code byte	data structure 500
510	flag bit	identification bit 510
520	repeat code	repeat data structure 525
530	count	repeat value
530	count	repeat counter value
530	count	remaining bits 510
550	data code	line number data structure 550
610	decimal values	uncompressed data stream 1000
640	binary code	compressed data stream 1020
700	decode table	decompression lookup table 700
720	red values	red scale illumination intensity value 710
730	green values	green scale illumination intensity value 720
740	blue values	blue scale illumination intensity value 730
	machine	system

DESCRIPTION OF THE INVENTION

Fig 1–Compression and Decompression Steps

Fig 1 illustrates a sequence of compression steps 100 and a sequence of decompression steps 150 of the present invention. The compression steps 100 comprise a sub-sampling step 110, a code lookup step 120 and a run-length encoding step 130. After completion of the compression steps 100, a stream of encoded data 140 is output to either a storage medium or a transmission channel. The decompression steps 150 comprise a run-length expansion step 160 wherein the stream of encoded data 140 is processed, a value lookup step 170 and an image reconstitution step 180.

Figs 2A to 2H Selecting Pixel Values for Encoding

Figs 2A to 2G illustrate alternatives for selecting a pixel value for encoding. The sub-sampling step 110 (Fig 1) includes sub-sampling of an 8 bit pixel value to obtain a value for use in the subsequent code lookup step 120 (Fig 1).

Video digitizing hardware typical has the options of storing the pixel values as a 32 bit pixel value 200 or a 24 bit pixel value 210, shown in Fig 2A and Fig 2B, respectively. The 32 bit pixel value 200 is composed of a blue channel 202, a green channel 204, a red channel 206, and an alpha channel 208. Each channel contains of 8 bits and can represent 256 saturation levels for the

particular color channel. For each channel the saturation intensity value of zero represents the fully off state, and the saturation intensity value of "255" represents the fully on state. The 24 bit pixel value 210 is composed of a blue component 212, a green component 214, and a red component 216. There is no component for the alpha channel in the 24 bit pixel value 210. Regardless of the structure, the blue channel 202 is equivalent to the blue component 212, the green channel 204 is equivalent to the green component 214, and the red channel 206 is equivalent to the red component 216.

In the present invention, the 32 bit pixel value 200 alternative is preferred due to the consistent alignment of 32 bit values in most computer memories; however for simplicity of illustration the alpha channel 208 will be omitted in Fig 2C to 2G.

If the video signal is digitized in color, the three color components may have different values. For example in Fig 2C, a RGB averaging diagram 220 illustrates a blue value 222 of 35 decimal, a green value 224 of 15, and a red value 226 of 10. One alternative is to sub sample from 24 bits to 8 bits by averaging the three color values to obtain an averaged value 228 that, in this example, has the value of 20. (10+15+35)/3 = 20

Fig 2D illustrates another alternative for selecting an 8 bit value in a blue selection diagram 230. In this example, a blue instance 232 has the value of 35, a green instance 234 has the value of 15, and a red instance 236 has the value of 10. In this alternative the blue instance 232 is always selected as a selected blue value 240.

Fig 2E illustrates another alternative for selecting an 8 bit value in a green selection diagram 250. In this alternative the green instance 234 is always selected as a selected green value 260.

Fig 2F illustrates another alternative for selecting an 8 bit value in a red selection diagram 270. In this alternative the red instance 236 is always selected as a selected red value 280.

If the video signal being digitized is grayscale, the three color components will have the same values. For example, in Fig 2G, a grayscale pixel 290 comprises a grayscale blue 292 with a value of decimal 40, a grayscale green 294 with a value of 40, and a grayscale red with a value of 40. Because the values are all the same, it makes no difference which grayscale color component is selected, a selected grayscale value 298 will have the value of 40 in this example.

The preferred embodiment of this invention uses the low order byte of the pixel value, which is typically the blue component as shown in Fig 2D.

Fig 2H illustrates an 8 bit pixel value 299 which is selected by one of the alternatives described above. The 8 bit pixel value 299 is equivalent to items referenced by numerals 228, 240, 260, 280, or 298. This reduction of the 32 bit pixel value 200 or the 24 bit pixel value 210

contributes a reduction in data size of 4:1 or 3:1, respectively. This reduction recognizes that for some images, such as medical images or grayscale images, no relevant information is lost.

Fig 3A–Encode Table

Fig 3A illustrates an encode table 300 of the preferred embodiment of the present invention. The encode table 300 may be implemented in the C programming language, as shown, and is an array of 256 bytes. These bytes are the codes 310. Each line of the array elements is documented with one of a series of line comments 320. The line comments 320 contain a column of minimum values 330, a column of maximum values 340, and a column of stepped values 360. The first column of encode table 300 is a column of line numbers 370. The encode table is arranged such that each line contains codes 310 that have a value equal to its line number.

The encode table 300 reduces the 8 bit value 299 to a 5 bit code. This reduction recognizes that for some images, such as medical images, no relevant information was lost. This reduction also eliminates noise from the video signal and thereby increases the efficiency of the run-length encoding step 130 (Fig 1).

Fig 3B–Chart of Values

Fig 3B, a chart of values 350, enumerates the range that the 8 bit pixel value 299 can have and the respective placement of each value in the encode table 300. The first column of the chart enumerates the line numbers 370. Column A contains the minimum values 330 for each line. Each row of the chart has from four to nine sequential entries. The last column of the chart, column J, contains the stepped values (360) that are used in the value lookup step 170 (Fig 1).

Fig 4A–Encode Flowchart

Fig 4A illustrates the encode flowchart 400 which represents the details of the preferred embodiment of the code lookup step 120 (Fig 1) and the run-length encoding step 130 (Fig 1) for the present invention.

The encoding begins at an encode entry 402. In a encode initialization step 403, a prior value P is set to a known value, preferably decimal "255" or hexadecimal 0xFF, a repeat count C is set to zero, an encoded length L is set to 0, and a completion flag "Done" is set to a logical value of false. Next, a get pixel step 404 obtains a pixel from the image being encoded. At a get value step 405, a value V is set to the 8 bit pixel 299 as derived from the pixel using one of the methods shown in Fig 2C to 2G, preferably the fastest as explained above. At a lookup encoded value step 406, an encoded value E is set to the value of one of the codes 310 (Fig 3) of the encode table 300 as indexed by V. Next, a compare previous 408 decision is made by comparing the values of E and P. If the

values are the same, a increment counter step 410 is executed and flow continues to the get pixel step 404 that obtains the next pixel from the image.

If the encode value E does not match the prior value P, then a check count overflow 412 decision is made. If the counter C is less than or equal to 128, then a new code step 414 is executed, otherwise a counter overflow step 420 is executed.

At step 414, the counter C is bit-wise AND-ed with hexadecimal 0x80 which sets the high order bit to a binary value of 1 and is placed in the encoded data 140 buffer A at the next available location as indexed by the encoded length L. Then, continuing inside flowchart step 414, L is incremented, the prior value P is placed in the encoded data 140 buffer A, L is incremented, the repeat count C is set to 1 and the prior value P is set to the encode value E. After step 414, a check end of data decision is made by checking to see if there are any more pixels in the image and if not if the last value is has been processed. Because this method utilizes a read ahead technique, step 414 must be execute one more time after the end of data is reached to process the last run-length. If there is more data in the image, flow continues to a check of the completion flag "Done" at step 422. If the check indicates that the process is not completed, flow continues to step 404.

If the end of data is reached but the completion flag "Done" is still false, flow continues to a set done step 418. At step 418, the completion flag "Done" is set to logical true, and flow continues to decision 412 where the last run-length will be output and flow will eventually exit through step 414, decision 416, decision 422, and then terminate at encode exit 428.

It is possible for the repeat count C to become larger than 128 requiring more bits than allocated by this method. This situation is handled by making the check count overflow 412 decision and executing the counter overflow step 420. At step 420, the value hexadecimal 0x80 is placed in the encoded data 140 buffer A at the next available location as indexed by the encoded length L. Then, continuing inside flowchart step 420, L is incremented, the prior value P is placed in the encoded data 140 buffer A, L is incremented, the repeat count C is decrement by 128. After step 420, flows continues to the check count overflow 412 decision. Thus when the encode value E repeats more that 128 times, multiple sets of repeat counts and encoded values are output to the encoded data 140 buffer.

This entire process is repeated for each image or video frame and the encoded length L is transmitted with the encoded data associated with each frame. The encoded length varies from frame to frame depending on the content of the image being encoded.

Fig 4B–Image and Pixel Stream

Fig 4B illustrates an image and its corresponding stream of pixels. A rectangular image 430 is composed of rows and columns of pixels. The image 430 has a width 440 and a height 450, both measured in pixels. Pixels in a row are accessed from left to right. Rows are accessed from top to bottom. Some pixels in the image are labeled from A to Z. Pixel A is the first pixel and pixel Z is the last pixel. Scanning left to right and top to bottom will produce a pixel stream 460. In the pixel stream 460, pixels A and B are adjacent. Also pixels N and O are adjacent even though they appear on different rows in the image. If adjacent pixels have the same code the process in Fig 4A will consider them in the same run.

Because the video signal being digitized is analog there will be some loss of information in the analog to digital conversion. The video digitizing hardware can be configured to sample the analog data into the image 430 with almost any width 440 and any height 450. The present invention achieves most of its effective compression by sub-sampling the data image with the width 440 value less than the conventional 640 and the height 450 value less than the convention 480. The preferred embodiment of the invention for use in a medical application with T1 Internet transmission bandwidth is to sample at 320 by 240. However, a sampling resolution of 80 by 60 may be suitable for some video application.

Figs 5A to 5C–Run-length Encoding Formats

Figs 5A to 5C show the formats for the run-length encoding. Fig 5A shows a code byte 500, with its high order bit designated as a flag bit 510.

Fig 5B shows a repeat code 520 comprising a Boolean value one in its flag bit 510 and a 7 bit count 530 in the remaining 7 low order bits. The seven bit count 530 can represent 128 values with a zero representing "128" and 1 through 127 being their own value.

Fig 5C shows a data code 550 comprising:

1. a Boolean value zero in its flag bit 510
2. two unused data bits: data bit 6 reference by 565 and data bit 5 reference by 570, and
3. five bits, data bits 4 to 0, referenced by 575, 580, 585, 590, and 595, respectively.

The five bits hold a 5 bit code selected from the codes 310 (Fig 3A) in the encode table 300 (Fig 3A).

The preferred embodiment of this invention uses the high order bit of the code byte 500 as the flag bit 510, because it results in the faster execution of the process. However, any bit could have been designated as the flag bit 510 with the same logical result.

Fig 6–Encoded Data Stream

Fig 6 shows a series of decimal values 610 comprising a first value 620 equal to decimal 0, a second value 622 equal to 0, a third value 624 equal to 0, a fourth value 626 equal to 0, a fifth value 628 equal to 0, a sixth value 630 equal to 2, and a seventh value 632 equal to 10. After the run-length encoding step 130 (Fig 1), the corresponding encoded data 140 (Fig 1) would be compressed down to four bytes of binary code 640 comprising a first byte 650 containing a repeat count 651, a second byte 652 containing a first code 653, a third byte 654 containing a second code 655, and a fourth byte 656 containing a third code 657. The repeat count 651 has a binary value of "0000101" which equals decimal five representing the run-length of the repeating value in the first five of the decimal values 610. The first code 653 has a binary value of "00000" which equals the repeated decimal value zero. The second code 655 has a binary value of "00010" which equals the non-repeated decimal value two. The third code 657 has a binary value of "01010" which equals the non-repeated decimal value ten.

Fig 7–Decode Table

Fig 7 illustrates a decode table 700 of the preferred embodiment of the present invention. The decode table 700 may be implemented in the C programming language, as shown, <u>or any programming language,</u> and is an array of 32 element with each element being a 32 bit pixel value 200 (Fig 2A). The decode table is comprised of a column of alpha values 710, a column of red values 720, and a column of green values 730, and a column of blue values 740, where the alpha values 710 are shifted by 24 bits, the red values 720 are shifted by 16 bits, and the green values 730 are shifted by 8 bits leaving the blue values 740 in place. The line in the decode table 700 contains one element. Each element comprising:

1. the alpha channel 208 (Fig 2A) with full intensity represented by hexadecimal 0xFF
2. the red channel 206 (Fig 2A)
3. the green channel 204 (Fig 2A)
4. the blue channel 202 (Fig 2A)

where the values of the three color channels are equal to the corresponding stepped values 360 (Figs 3A and 3B) associated with each line of the encode table 300 (Fig 3A).

Although each element is documented as an expression composed of various bit shift and bit-wise OR operations, the expression is evaluated by the compiler when the program is compiled so that each element of the decode table 700 is a 32 bit pixel value 200 (Fig 3A) ready for direct placement in the decompressed image.

In the preferred embodiment of this invention the 32 bit pixel value 200 (Fig 2A) is used; however other embodiments use the 24 bit pixel value 210 (Fig 2B) or other pixel sizes known in the art. Embodiments of other common pixel depths of 16 bit, 15 bit, 8 bit, 4 bit, 3 bit or 1 bit could be used without limiting the scope of this invention. Of course any source resolution less than or equal to five bits would benefit from a modified and shortened encode and decode table or could be encoded by shifting bits as explained below.

Fig 8–Alternate Code Selection

Fig 8 illustrates an alternate embodiment of this invention where tables are not used for encoding and decoding. Instead, the high order 5 bits 810-818 of an 8 bit pixel 800 are shifted to the right by 3 bit positions to form a five bit sample 830. The upper 3 bits of 830 are ignored bits 850. This shifting to obtain a five bit code replaces steps 405 and 406 of the flowchart in Fig 4A. The same run-length encoding method is used. During decompression the five bit sample 830 is shifted to the left by 3 bit positions to form a decompressed pixel 860, where the 3 low order bits 880 are filled with zero binary values.

Fig 9–Decode Flowchart

Fig 9 illustrates the decode flowchart which presents the details of the preferred embodiment of the value lookup step 170 (Fig 1) and the image reconstitution step 180 (Fig 1).

The decoding ~~beings~~ begins at a decode entry 900. In a decode initialization step 901, a repeat counter C is set to one, an encoded length L is set to the value obtained with the encoded data 140 (Fig 1), and an index I is set to 0. Next, a get code step 902 obtains a signed byte X from the encoded data 140 (Fig 1) array A. A determine type 906 decision checks to see if the signed byte X is less than 0.

If the signed byte X is less than zero, it is because the high order bit, the flag bit 510 (Fig 5A) is set to binary value 1, as in Fig 5B, indicating that the byte X is a repeat code 520. Flow goes to assign counter step 912 where the count 530 (Fig 5B) is extracted from X and placed in the repeat counter C and the next code is accessed by incrementing the index I and returning to the get code step 902.

If the signed byte X is greater than or equal to zero, it is because the flag bit 510 (Fig 5A) is set to binary 0, as in Fig 5C, indicating that the byte X is a data code 550. Flow goes to a decode lookup step 908 where the value of byte X is used to index into the decode table 700 (Fig 7) to obtain a pixel value V. Flow continues to a check zero count 909 decision.

The 909 decision always fails the first time ensuring that a place pixel step 910 is executed. The place pixel step 910 places the pixel value V in the next location of the decompressed image and decrements the repeat counter C and returns to the 909 decision. The pixel value V is placed repeatedly until C decrements to zero. Then the 909 decision branches flow to a reset counter step 914. At step 914 the repeat counter is reset to 1 and the index is incremented to select the next code.

Flow continues to the check length 916 decision where the index I is compared to the encoded length L to determine if there are more codes to be processed. If I is less than L flow returns to step 902, otherwise the decode process terminates at a decode exit 918.

The entire decode process is repeated for each frame image.

Figs 10A to 10C–Encrytpion Key, Encryption Table, and Decryption Table

Fig 10A shows an encryption key 1000. The first column shows the original code. The second column shows the corresponding cipher code.

Fig 10B shows an encryption table 1010, and Fig 10C shows a decryption table 1020. The encryption table 1010 has the same format as the standard encode table 300 (Fig 3A), and the decryption table 1020 has the same format as the decode table 700 (Fig 7). However the entries in both tables are rearranged such that the direct correlation between the intensity level and the position in the table is broken. When these versions of the tables are used, the encode and decode processes and their speed of execution are substantially the same but the encoded data 140 (Fig 1) becomes a cipher and has a higher level of security. It should be recognized by one with ordinarily skill in the art that there are other embodiments of the present invention with different encryption/decryption table rearrangements.

Advantages

Filtering and Image Enhancement

The stepped values 360 (Fig 3A) are a significant discovery of the present invention. The use of these specific values results in high quality decompressed images when the original image is generated by an electronic sensing device such as an ultrasound machine. The encode table 300 is arranged such that the spikes in the video signal are filtered in the high and low end, line numbers 31 and 0, respectively. The remaining values are distributed more evenly with larger ranges at line 3, 7, 15, 19, 23, and 27.

By altering the contents of the encode table 300 and the decode table 700 various filters can be implemented to enhance the image quality. A high or low noise filter can be beneficial when the image is generated by an imaging technology such as radar, ultrasound, x-ray, magnetic resonance,

or similar technology. Variations in the encode and decode table can be made to enhance the perceived quality of the decompressed image. Therefore, altering the contents, shape, or size of the encode table 300 and the decode table 700 is anticipated by this invention and specific values in the tables should not be construed as limiting the scope of this invention.

Execution Speed

The preferred embodiment of this invention use a number of techniques to reduce the time required to compress and decompress the data.

The methods require only a single sequential pass through the data. Both the compression steps 100 and the decompression steps 150 access a pixel once and perform all calculations.

When selecting the 8 bit pixel value 299, the preferred embodiment selects the low order bits from the 32 bit pixel value 200 or the 24 bit pixel value 210 so that an additional shift operation is avoided.

The encode table 300 is a fast and efficient way to convert the 8 bit pixel value 299 into one of the 5 bit codes 310.

The decode table 700 contains 32 entries each comprised of the 32 bit pixel value 200 that are ready for placement in the decompressed image. Although each element is documented as an expression composed of various bit shift and bit-wise OR operations, the expression may also be evaluated by a compiler when the program is compiled so that each element of the decode table 700 becomes a 32 bit pixel value 200.

General Purpose

Although the preferred embodiment of the present invention is tuned to the characteristics of a medical image, its lossless compression of the sampled data results in high quality video streams that have general purpose application in a number of areas including, without limitation, video conferencing, surveillance, manufacturing, and rich media advertising.

Lossless Nature/No Artifacts

Once the analog signal is sub-sampled and filtered to select a five bit code value which eliminates some of the real world defects, the methods of the present invention compress and decompress the data with no irreversible data loss. Unlike JPEG and MPEG, the decompressed image never suffers from artificially induced blocking or smearing or other artifacts that are result of the lossy compression algorithm itself. As a result even a small sub-sample of the image remains clear and true to the perceived quality of the original image.

Conclusion, Ramification, and Scope

Accordingly, the reader will see that the compression and decompression steps of the present invention provides a means of digitally compressing a video signal in real time, communicating the encoded data stream over a transmission channel, and decoding each frame and displaying the decompressed video frames in real time.

Furthermore, the present invention has additional advantages in that:

1. it provides a means of filtering real world defects from the video image and enhancing the image quality;
2. it allows for execution of both the compression and decompression steps using software running on commonly available computers without special compression or decompression hardware;
3. it provides decompressed images that have high spatial quality that are not distorted by artifacts of the compression algorithms being used;
4. is provides a scalable means of video compression; and
5. it provides a means for reducing the space required in a storage medium.

Although the descriptions above contain many specifics, these should not be construed as limiting the scope of the invention but as merely providing illustrations of some of the preferred embodiments of this invention. For example, stepped values in the encode and decode tables can be altered and the same relative operation, relative performance, and relative perceived image quality will result. Also, these processes can each be implemented as a hardware apparatus that will improve the performance significantly.

Thus the scope of the invention should be determined by the appended claims and their legal equivalents, and not solely by the examples given.

Claims: We claim:

1. A method of compression of graphic images which make up a video stream, comprising the steps of:
   (a) sub-sampling pixels from an image selected from said graphic images;
   (b) selecting a code based on a number of bits from each pixel selected from said pixels;
   (c) run-length encoding repeated instances of said code;
   (d) repeating steps (b) and (c) until each said pixel is encoded in an encoded data buffer; and
   (e) streaming said buffer which represents said graphic images.
2. The method of claim 1 wherein the rate of sub-sampling frames is greater than or equal to 15.
3. The method of claim 1 wherein image dimensions are less than or equal to 320 by 240.
4. The method of claim 1 wherein said number of bits is five and said code is determined by extracting the five most significant bits from each pixel.
5. The method of claim 1 wherein said number of bits is five and said code is obtained from an encode table.
6. <u>The method of claim 1 wherein an</u> ~~An~~ encoded video signal ~~comprising~~ <u>comprises</u> a series of said encoded data buffers.
7. A storage medium in which the encoded video signal as claimed in claim 6 is stored.
8. A method of decompressing an encoded video signal, comprising the steps of:
   (a) reading a stream of run-length encoded codes;
   (b) determining a series of pixels based on the values and run-lengths of said codes;
   (c) combining said pixels into an image; and
   (d) displaying a series of said images.
9. The method of claim 8 wherein the display frame rate is greater than or equal to 15.
10. The method of claim 8 wherein the width and the height of said image are less than or equal to 320 by 240, respectively.
11. The method of claim 8 wherein said codes are five bits in length and said pixel's values are determined by using the least significant bits of said codes as the five most significant bits of each pixel.
12. The method of claim 8 wherein each of said pixel values are obtained from a decode table, whereby said image is an enhanced representation of the original image.

13. The method of claim 5 wherein the lines of said encode table are randomly ordered forming an encryption table so that the direct correlation between the original values and their representative codes are encrypted.

14. The method of claim 12 wherein the lines of the decode table are ordered in a sequence matching said encryption table so that the correct final pixel values are displayed.

15. A machine for compressing of a plurality of video frames which make up a video signal, comprising

    (a) a video digitizer configured to digitizing a frame from said video frames;

    (b) a video memory which is able to receive a plurality of pixels from said video digitizer;

    (c) run-length encoding circuit for counting repeated instances of a pixel value when scanning said plurality of pixels and output a series of run-lengths and code values as encoded data;

    (d) a memory which is able to store said encoded data;

    (e) an input/output device.

16. The machine of claim 15 wherein said run-length encoding circuit performs a table lookup to translate said pixel values into encrypted enhancement codes.

17. The machine of claim 15 wherein said input/output device is a storage medium.

18. The machine of claim 15 wherein said input/output device is a communications transmission channel.

19. A machine for decompressing an stream of encoded data that represents a video signal, comprising:

    (a) an input/output device for reading said stream of encoded data;

    (b) a run-length decoding circuit which can decode the encoded data and output a stream of pixel values; and

    (c) a memory that is able to store an image comprising said stream of pixel values that can be displayed as frames of a video sequence.

20. The machine of claim 19 wherein said run-length decoding circuit performs a decode table lookup.

21. A method of compressing a stream of data representing a stream of pixels, each pixel having a corresponding illumination intensity value, the method comprising the steps of:
    a. matching the illumination intensity value representing a pixel with a current line number;
    b. determining if the current line number matches a previous line number of an immediately prior pixel;
    c. incrementing a repeat counter if the current line number matches the previous line number;
    d. encoding a repeat data structure with the repeat counter, if the current line number does not match the previous line number and the repeat counter has a value greater than zero; and
    e. encoding a line number data structure with the current line number if the current line number does not match the previous line number;

    wherein a compressed stream of data is formed from combinations of the line number data structure and the repeat data structure.
22. The method according to claim 21 further comprising the step of resetting the repeat counter to zero after the repeat data structure is encoded.
23. The method according to claim 21 wherein the repeat data structure and the line number data structure include an identification bit,

    wherein when the identification bit is in a first state, a repeat data structure is encoded and when the identification bit is in a second state, a line number data structure is encoded.
24. The method according to claim 23 further comprising the steps of:
    a. receiving the compressed stream of data, one data structure at a time;
    b. reading the identification bit within the data structure to determine if the data structure is a line number data structure or a repeat data structure;
    c. generating a representative average illumination intensity value corresponding to the line number if the data structure is a line number data structure; and
    d. generating a number of representative average illumination intensity values corresponding to the line number of a last received line number data structure if the data structure is a repeat data structure, wherein the number is equal to the repeat counter within the repeat data structure.

Abstract: Methods, medium, and machines which compress, encode, enhance, transmit, decompress and display digital video images in real time. Real time compression is achieved by sub-sampling each frame of a video signal, encoding and filtering the pixel values to codes, and run-length encoding the codes. Real time transmission is achieved due to high levels of effective compression. Real time decompression is achieved by decoding and decompressing the encoded data to display high quality images. High levels of effective compression also reduce the storage space requirement for recorded video.